Tune to Learn: How Controller Gains Shape Robot Policy Learning

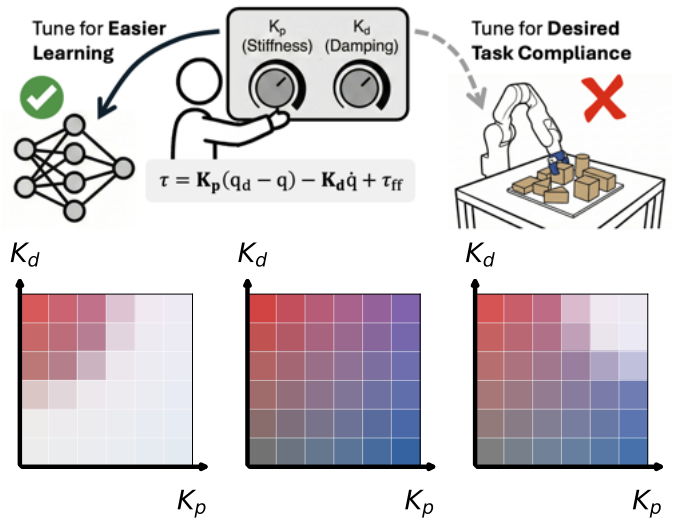
Author Names Omitted for Anonymous Review.

***Abstract*—Position controllers have become the dominant interface for executing learned manipulation policies. Yet a critical design decision remains understudied: how should we choose controller gains for policy learning? The conventional wisdom is to select gains based on desired task compliance or stiffness. However, this logic breaks down when controllers are paired with state-conditioned policies: effective stiffness emerges from the interplay between learned reactions and control dynamics, not from gains alone. We argue that gain selection should instead be guided by learnability: how amenable different gain settings are to the learning algorithm in use. In this work, we systematically investigate how position controller gains affect three core components of modern robot learning pipelines: behavior cloning, reinforcement learning from scratch, and sim-to-real transfer. Through extensive experiments across multiple tasks and robot embodiments, we find that: (1) behavior cloning benefits from compliant and overdamped gain regimes, (2) reinforcement learning can succeed across all gain regimes given compatible hyperparameter tuning, and (3) sim-to-real transfer is harmed by stiff and overdamped gain regimes. These findings reveal that optimal gain selection depends not on the desired task behavior, but on the learning paradigm employed. As pipelines increasingly combine imitation learning with reinforcement learning fine-tuning and sim-to-real transfer, understanding how this shared interface layer interacts with each paradigm becomes critical for principled system design.**

I. INTRODUCTION

Position controllers are rapidly becoming the de facto choice for low-level control in robot learning. Their wide hardware support and intuitive nature have made them the dominant interface for executing learned manipulation policies. Yet while classical control theory provides clear guidance on selecting gains to achieve desired tracking bandwidth, disturbance rejection, or impedance characteristics, no analogous principles exist for the learning setting. An important design decision remains overlooked: how should we choose controller gains when *learning* data-driven manipulation policies?

The standard approach treats gain selection as a problem of achieving desired task behavior—contact-rich manipulation calls for low stiffness and high damping to better comply with unexpected contacts, while precision tasks call for high stiffness and low damping to accurately track position commands. But this framing conflates two distinct roles that position controllers play. When tracking open-loop trajectories, the controller *is* the *behavior*—gains directly determine how the robot responds. When paired with a learned policy, however, the controller becomes an *interface* between the policy and the physical world. The policy learns through this interface during training and acts through this interface at deployment. Viewed this way, gains function less as behavioral parameters and more as an *inductive bias*—an implicit prior over the space of closed-loop behaviors that shapes what the policy can easily express and learn.

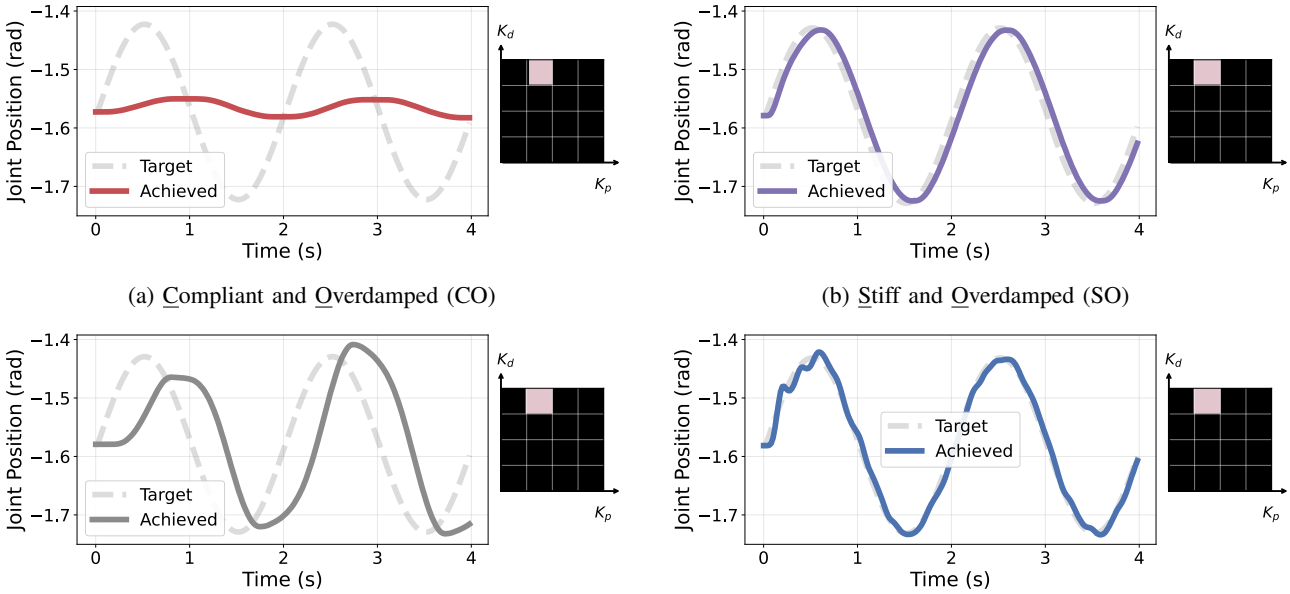


(a) BC (b) RL (c) Sim2Real

Fig. 1: **Different robot learning paradigms prefer different controller gain interfaces.** Contrary to conventional wisdom of tuning gains for desired task compliance, optimal gains depend on the learning paradigm. Based on our experimental findings, heatmaps illustrate representative gain preferences for (a) behavior cloning, which favors compliant, overdamped gains, (b) reinforcement learning, which adapts to nearly any setting, and (c) sim-to-real transfer, which is degraded by stiff and overdamped gains.

This distinction matters because learned policies are reactive: they observe the current state and output corrective commands. A policy can achieve arbitrarily stiff or compliant task-level behavior regardless of the underlying joint-level gains, simply by modulating the magnitude and timing of its outputs. The gains, therefore, do not constrain what behaviors are reachable. We hypothesize that the gains, instead, constrain the learning problem: how easy it is to fit action labels and how errors compound during closed-loop execution, which training configurations yield successful RL policies, and whether modeling discrepancies amplify into instability during sim-to-real transfer.

Once we recognize controller gains as learning interface parameters rather than behavioral parameters, the design question becomes: which interface properties facilitate learning?



(a) Compliant and Overdamped (CO)

(b) Stiff and Overdamped (SO)

(c) Compliant and Underdamped (CU)

(d) Stiff and Underdamped (SU)

Fig. 2: **Controller gains induce diverse action–response dynamics.** We evaluate a broad range of representative gain configurations and their resulting dynamic responses to assess their impact on learnability.

And critically, do different learning paradigms prefer different interfaces, serving as a conducive *inductive bias*? We investigate these questions systematically across three paradigms of modern robot learning and present the following findings:

1) Behavior cloning (BC) performs best with *compliant* and *overdamped* gains. Due to its inherent error dampening properties, action labels generated under compliant gain regimes (Sec. IV-A for experiment design and Sec. V-A for results)
2) Reinforcement learning (RL) from scratch is agnostic to gain setting, as long as the remaining hyperparameters are tuned to be compatible with the given gain setting. We verify this by obtaining equivalently successful RL policies for all gain regimes across multiple manipulation and locomotion tasks. (Sec. IV-B for experiment design and Sec. V-B for results)
3) When transferring policies from simulation to the real-world, *stiff* and *overdamped* controllers exacerbate the motor-level sim-to-real gap. (Sec. IV-C for experiment design and Sec. V-C for results)

Our findings converge on a unified picture of how controller gains shape learning, providing both conceptual clarity and practical guidance for this widely used yet underexplored design decision.

II. Related Works

Position and Impedance Control. PD control with gravity compensation [1] is the dominant low-level interface in robot learning: policies output joint position targets $\mathbf{q}_d$ tracked by $\boldsymbol{\tau} = \mathbf{K}_p(\mathbf{q}_d - \mathbf{q}) + \mathbf{K}_d(\dot{\mathbf{q}}_d - \dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q})$, where the gain matrices $\mathbf{K}_p, \mathbf{K}_d$ determine joint stiffness and damping. Despite well-established stability theory [2], gain selection in practice remains largely heuristic.

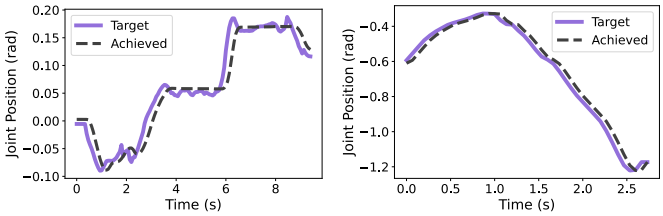
Low-Level Control in Robot Learning. Position-controlled action spaces convert policy outputs to torques via feedback laws, making gains an implicit part of action space design. Prior work has compared action representations [3, 4] and framed action space selection as an inductive bias for locomotion [5], but without varying gains within a given representation. Several works learn variable impedance policies [6, 7, 8]; notably, Margolis *et al.* [8] show that task-level compliance is dictated by training incentives rather than PD gains. We build on this insight: rather than learning compliance, we study how *fixed* gains shape the learning process. On the sim-to-real side, domain randomization [9, 10] commonly perturbs gains and motor parameters [11, 12], yet how the *nominal* gain setting affects transfer remains unexplored.

Gain Settings in Large-Scale Datasets.

Controller gain configurations in large-scale datasets are rarely documented. Analyzing DROID [13] and Open X-Embodiment [14] datasets, we find that achieved positions closely track commands with minimal lag and overshoot (Fig. 3), suggesting stiff gains have become an implicit default.

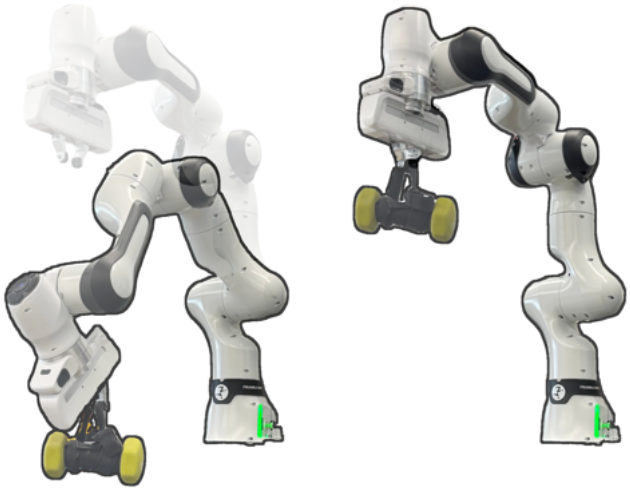
III. Decoupling Gains from Task Compliance

In this section, we validate a central claim: *a closed-loop policy can realize arbitrary task-level impedance, independent of the low-level controller gains.* We demonstrate this through two counterintuitive pairings using RL policies trained to maintain a fixed pose under external disturbances, with a distance-based reward $r(\mathbf{q}) = 1 - \tanh(\|\mathbf{q} - \mathbf{g}\|^2/\lambda)$.



(a) DROID (b) RT-X NYU Franka Play

Fig. 3: Tracking response curves from existing robot datasets reveal tight command-following behavior, suggesting stiff controller gains are prevalent in existing data collection pipelines.



(a) Compliance w/ stiff gain (b) Stiffness w/ compliant gain

Fig. 4: **Task-level impedance can be decoupled from low-level controller gains with learned policies.** A learned policy can achieve (a) *compliant* behavior despite stiff low-level gains, and (b) *stiff* behavior despite compliant gains.

Stiff behavior with compliant gains. Despite compliant low-level gains, a small λ strongly penalizes deviations from the goal, encouraging the policy to actively counteract disturbances (Fig. 4b).

Compliant behavior with stiff gains. A large λ combined with an action smoothness penalty $\alpha\|\Delta a_t\|^2$ encourages the policy to yield smoothly under disturbances even with stiff gains (Fig. 4a).

IV. Experiments

Now that we have established that gains of underlying position controllers do not dictate the task-level behavior (and thus should not be tuned for it), we aim to answer: *how do gains affect the learning process?* In this section, we present the experiment protocols we devised to study this problem systematically.

A. Behavior Cloning (BC)

For behavior cloning, we investigate how gain setting $\mathbf{K}$ affects (1) closed-loop BC policy performance and (2) teleoperation experience during data collection.

Examining Gain-Dependent BC Performance. To isolate the effect of gains on learning, we need datasets where gains affect only the actions, not the state trajectories. Collecting demonstrations independently per gain setting confounds gain-dependent actions with different state distributions. We address this via Torque-to-Position Retargeting (TPR): we first generate demonstrations at 500Hz using torque commands, then retarget to position targets for each $(\mathbf{K}_p, \mathbf{K}_d)$:

$$\mathbf{q}_{\text{des}}(t) = \mathbf{q}(t) + \mathbf{K}_p^{-1}\left(\boldsymbol{\tau}(t) + \mathbf{K}_d\dot{\mathbf{q}}(t)\right), \tag{1}$$

where $\tau(t), q(t), \dot{q}(t)$ are from the original torque demonstration. Retargeted commands are replayed at 50Hz, keeping only successful rollouts. This yields datasets $\mathcal{D}(s, a(\mathbf{K}))$ with nearly identical state trajectories, isolating gain-dependent actions as the sole variable. We conduct this process in simulation.

We train BC policies for each gain configuration. Our nominal setup uses a VAE with MLP, observation history length 10, and action chunk size 10, with privileged simulation states as inputs and absolute joint-space actions as outputs. We verify gain preferences are consistent across architectures (MLP vs. Transformer), model classes (regression, VAE, diffusion [15]), temporal structure, input modalities, and output representations; full ablations are in Appendix A-A.

Studying How Gain Affects Teleoperation. We conducted a user study examining how gains affect human teleoperation. Since the input mapping $\phi(\mathbf{u}, \mathbf{x}) \to \mathbf{x}_{\text{des}}$ modulates how the robot feels, we allow each participant to tune ϕ per gain setting before evaluation:

$$\phi^\star(\mathbf{K}) = \arg\max_{\phi} \mathcal{Q}(\phi; \mathbf{K}), \tag{2}$$

where $\mathcal{Q}$ denotes perceived control quality. Participants teleoperated a Franka Research 3 via SpaceMouse on a non-prehensile box manipulation task (Fig. A-12) with randomly ordered, blinded gain configurations. We recorded completion time, success/failure, and subjective 1–5 ratings across 500 trials.

B. Reinforcement Learning

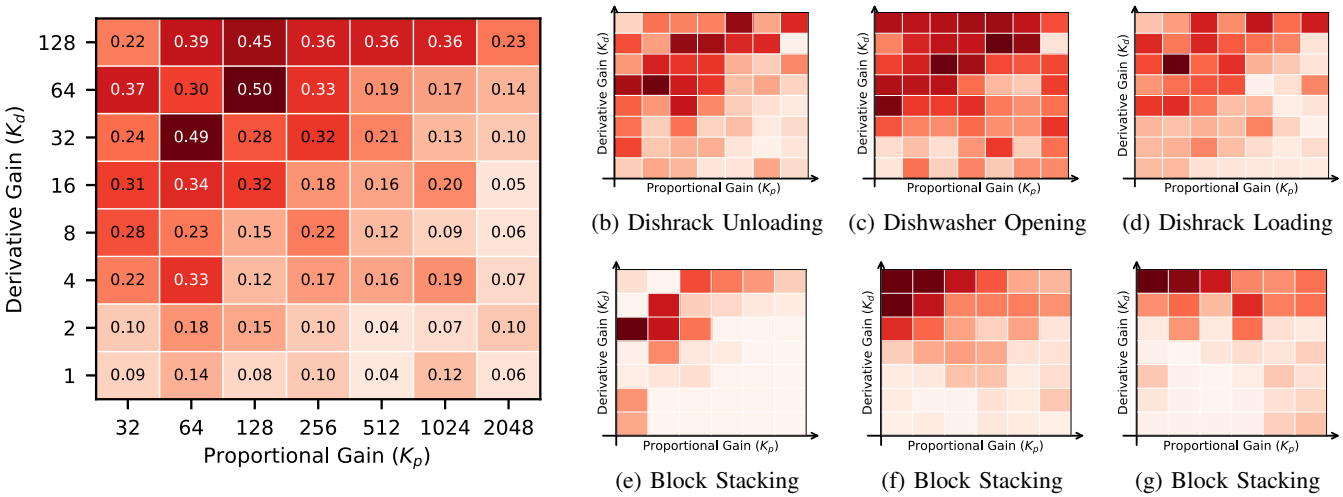
Can RL discover solutions regardless of gain settings? RL performance is sensitive to hyperparameters, so we must avoid conflating gain effects with suboptimal configurations. Following *environment shaping* [16], we re-tune per-joint action scales and reward weights for each gain setting via hyperparameter optimization [17]:

$$h^\star(\mathbf{K}) = \arg\max_{h} \; J\big(\pi^\star(h; \mathbf{K})\big), \tag{3}$$

where $\pi^\star(h; \mathbf{K})$ is the converged policy under gains $\mathbf{K}$ and hyperparameters h[1].

Can gains ease hyperparameter tuning? We also investigate whether certain gain regimes yield larger, more easily

[1]We trained policies using the SKRL implementation [18] of PPO [19]. Tasks are modified from template tasks from IsaacLab [20].



(a) Bimanual Handover

(b) Dishrack Unloading (c) Dishwasher Opening (d) Dishrack Loading

(e) Block Stacking w/ Task Space Control (f) Block Stacking /w GravComp (g) Block Stacking /wo GravComp

Fig. 5: **behavior cloning prefers *compliant* and *overdamped* controller gains.** Closed-loop rollout success rates across a grid of proportional ($\mathbf{K}_p$) and derivative ($\mathbf{K}_d$) gains for diverse manipulation tasks and robot embodiments. Each heatmap reports success averaged over evaluation rollouts. Across tasks, higher success rate (darker red) consistently concentrates in the compliant, overdamped regime (upper-left), while stiff or weakly damped controllers yield degraded performance.

discoverable regions of successful hyperparameters. We record success rates across the hyperparameter landscape during 50 trials per gain setting, continuing all trials to completion.

C. Sim-to-Real

We examine whether certain gain settings transfer more reliably from simulation to real hardware, studying reaching tasks on a Franka Research 3.

Gain-Specific System Identification. For each gain configuration, we excite the real robot with sinusoidal targets and optimize simulation parameters ψ to match state trajectories:

$$\psi^\star(\mathbf{K}) = \arg\min_{\psi} \sum_{t=0}^{T} \|\mathbf{x}(t;\mathbf{K}) - \bar{\mathbf{x}}(t;\psi)\|^2 \tag{4}$$

where $\mathbf{x} = (\mathbf{q}, \dot{\mathbf{q}})$ is the real state and $\bar{\mathbf{x}}(\cdot;\psi)$ its simulated counterpart.

Gain-Dependent Sim-to-Real Transfer. For each gain setting, we train RL policies in the calibrated simulation, discovering transferable solutions via $h^\star(\mathbf{K}) = \arg\max_h \tilde{J}(\pi^\star(h;\mathbf{K}))$, where $\tilde{J}$ augments the objective with real-world limit penalties. Policies are deployed zero-shot. We also ablate with domain randomization (10% perturbation of system-identified parameters). We measure sim-to-real *trajectory error*:

$$\mathcal{E} = \underbrace{\|\mathbf{q}_{\text{sim}} - \mathbf{q}_{\text{real}}\|^2}_{\text{position error}} + \underbrace{\|\dot{\mathbf{q}}_{\text{sim}} - \dot{\mathbf{q}}_{\text{real}}\|^2}_{\text{velocity error}} \tag{5}$$

averaged over 30 real-world rollouts per gain setting.

V. Results

A. behavior cloning

> **Result V-A-I** (Learnability): behavior cloning strongly prefers *compliant* and *overdamped* gains (i.e., top left region of Fig. 2).

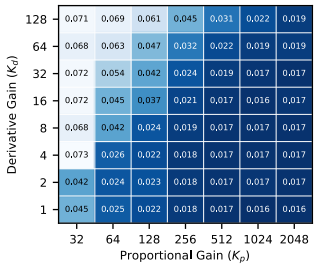
Figure 5 illustrates a consistent preference for *compliant* and *overdamped* gains across a broad grid of controller settings and diverse manipulation tasks. This trend persists across training configurations: state-based vs. image-based, action-chunked vs. non-chunked, with or without state histories (Appendix A-A), task-space vs. joint-space control (Fig. 5e), and with or without gravity compensation (Fig. 5g).

Higher MSE Loss, Better Performance. Policies with higher closed-loop performance often exhibit *higher* validation MSE loss (Fig. 6a), indicating that compliant-regime action targets are harder to fit. Yet these policies consistently outperform their low-loss counterparts during deployment.

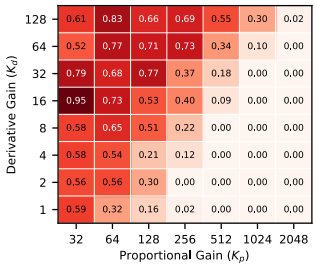
Compliant Controllers Attenuate Action Errors. The apparent paradox is explained by error-dampening properties of compliant controllers. We validate this by executing identical open-loop action sequences with injected noise across all gain configurations (Fig. 6b): compliant and overdamped gains maintain higher success rates under the same perturbations. For a given action prediction error, the robot moves *less* under compliant gains, preventing error accumulation.

> **Result V-A-II** (Effect on Data Collection): *Compliant* and *overdamped* gain settings yield comparable teleoperated data collection efficiency compared to other gain regimes, achieving similar yield and operator preference.

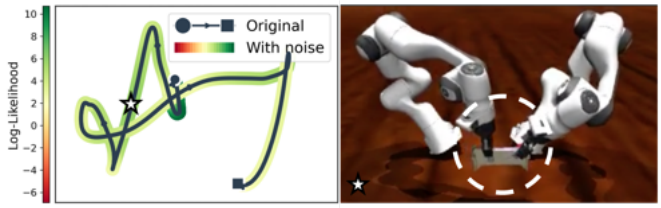
Our user study reveals that compliant, overdamped gains do not hinder teleoperation (Fig. 7): with optimized input shaping $\phi^\star(\mathbf{K})$ (Eq. 2), these settings achieve comparable or better success rates (Fig. 7a) and subjective ratings (Fig. 7c). Tuning



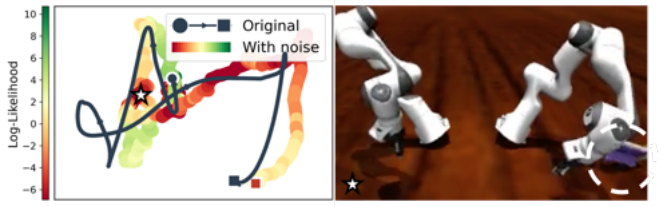
(a) Validation Loss



(b) Open-loop Success Rate with Action Noise

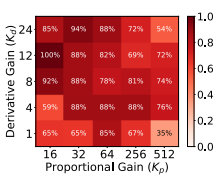


(c) Noised Open-loop Rollout for Compliant and Overdamped Gains

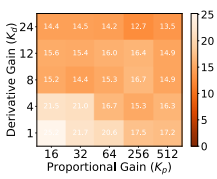


(d) Noised Open-loop Rollout for Stiff and Underdamped Gains

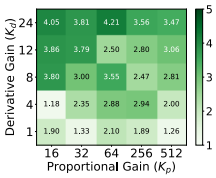
Fig. 6: **Compliant controllers attenuate action errors.** (a) Validation MSE loss during training: compliant gains yield higher loss, while stiff gains achieve lower loss. (b) Open-loop success rate under action noise: compliant gains maintain high success while stiff gains completely fail. (c) Compliant gains keep the perturbed trajectory close to the original, while (d) stiff gains cause large deviations that lead to task failure.



(a) Success Rate



(b) Completion Time



(c) User Rating

Fig. 7: **Teleoperation performance under different gain regimes.** With optimized input shaping $\phi^\star(\mathbf{K})$ (Eq. 2), compliant and overdamped controllers (grid top-left) achieve similar or better success rates, user ratings, and shorter completion time to stiffer settings.

the scaling factor α compensates for reduced responsiveness, so adopting compliant gains imposes no penalty on data collection.

B. Online Reinforcement Learning

> **Result V-B-I** (RL Solution Existence): Online reinforcement learning *can* discover behaviors regardless of gain setpoints.

Unlike BC, on-policy RL trains on self-generated data, allowing the policy to encounter and compensate for its own errors. We find that all gain regimes *can* yield working controllers given appropriate environment shaping (Table I).

> **Result V-B-II** (RL Environment Shaping Landscape): The gain setting modulates the hyperparameter optimization landscape, but no regime is consistently easier to optimize.

Fig. 8 visualizes the optimization landscape for three tasks. While gains clearly modulate the landscape, no regime is consistently easier: compliant, overdamped gains yield a wider successful region for FR3 Joint-Reach (Fig. 8a), while stiff, underdamped gains are least sensitive for FR3 Lift-Cube (Fig. 8b). This may reflect genuine task-dependence or artifacts of optimizing a low-dimensional hyperparameter slice.

C. Sim-to-Real

> **Result V-C-I** (System Identification): Black box parametric optimization based system identification prefers *stiff* and *overdamped* gains (i.e., upper right region of Fig. 2)

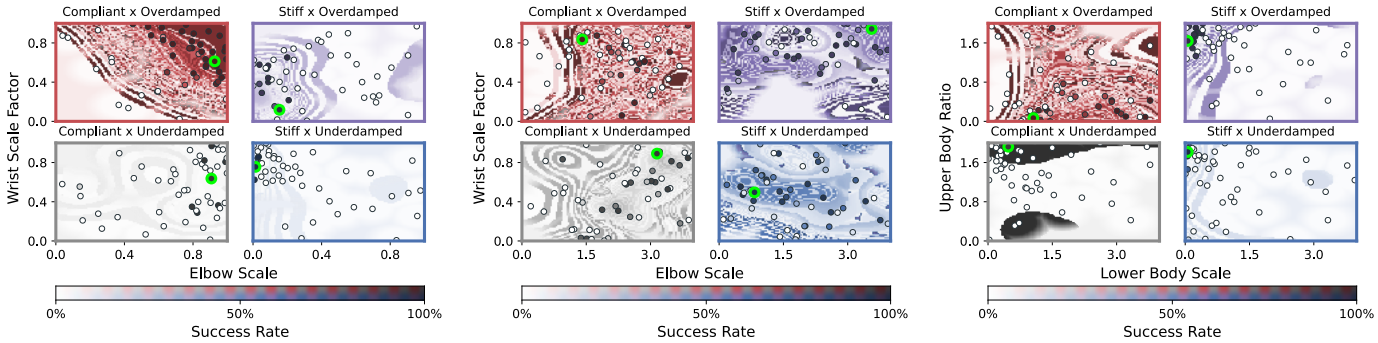
The MSE between simulated and real response curves after system identification, i.e., $\mathcal{S}^\star(\mathbf{K}) = \min_\psi \mathcal{S}(\mathbf{K}, \psi)$ (Eq. 4), is over an order of magnitude lower for the stiff, overdamped regime compared to other gain settings (Fig. 9a). We attribute this to stiff, overdamped dynamics filtering out nonlinearities that are difficult to capture with idealized simulated actuators: high damping suppresses high-frequency effects such as joint flexibility and transmission dynamics, while high stiffness reduces sensitivity to steady-state errors from imperfect gravity compensation or stiction.

> **Result V-C-II** (Sim2Real Transferability): Sim2Real transferability, however, is lower with *stiff* and *overdamped* gain setpoints.

Trajectory Error and Closed-Loop Amplification. Stiff and overdamped gains exhibit the worst sim-to-real trajectory error (Fig. 10), with the dominant failure mode being high-frequency oscillation that persists even with domain randomization (Fig. 10b). The instability emerges not from the

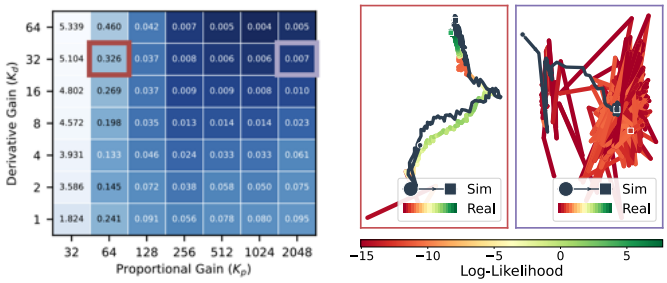
TABLE I: **RL solution existence across gain regimes.** For each task, we verify that at least one successful policy can be discovered in every gain regime given appropriate environment shaping. A checkmark indicates that gain regimes corresponding to four corner extremes in Fig. 2 yield working controllers. Videos of discovered behaviors for each gain settings are available on our project website.

Task / Platform	Existence Proof
FR3 Joint-Reach	✓
FR3 EE-Reach	✓
FR3 Lift Cube	✓
FR3 Open Drawer	✓
Unitree G1 Track Velocity	✓



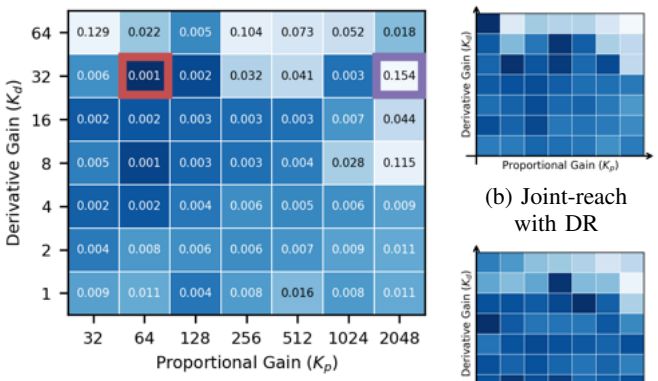
(a) FR3 Joint-Reach (b) FR3 Lift Cube (c) Unitree G1 Track Velocity

Fig. 8: **Success rate landscape during RL hyperparameter optimization for three tasks.** Across all tasks and gain regimes, we discover policies with 99%+ success rate (green circles). The size and shape of the successful region varies widely across gain settings and tasks.



(a) SysID Modeling Error (b) Real vs. Sim Rollouts

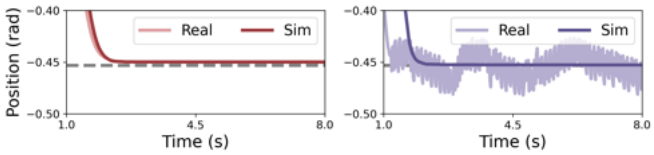
Fig. 9: **Stiff and overdamped gain settings yield lower SysID modeling errors, but exhibit larger closed-loop Sim2Real errors**. Policy observations during closed-loop rollout evolve similarly between sim and real (b-left) for compliant, overdamped gains, but very dissimilarly (b-right) for stiff, overdamped gains.



(a) Joint-Reach Sim2Real trajectory error without Domain Randomization

(b) Joint-reach with DR

(c) EE no DR



(d) Real vs. sim wrist joint trajectories.

Fig. 10: **Stiff and overdamped gain settings reduce sim2real transferability.** The Sim2Real trajectory error (Eq.5) is consistently larger (light blue) in the stiff and overdamped regime (a-c). The primary Sim2Real failure mode is high-frequency oscillation (d).

controller itself, but from its closed-loop interaction with the policy: stiff controllers aggressively track potentially erroneous commands, amplifying small modeling errors and pushing the policy into out-of-distribution states (Fig. 9b). This creates an inverse relationship between system identification accuracy and transfer quality—naively choosing gains that minimize modeling error can paradoxically increase sim-to-real error.

VI. Conclusion and Remarks

We have presented a systematic study of how position controller gains shape learning dynamics across three paradigms of modern robot learning. Our findings reveal that gains function not as behavioral parameters, but as an inductive bias that modulates the learning interface between policy and environment. behavior cloning favors compliant, overdamped regimes; reinforcement learning adapts to any gain setting given compatible hyperparameters; and sim-to-real transfer suffers with stiff, overdamped configurations. These results provide both conceptual clarity and practical guidance for a widely used yet underexplored design decision.

As modern robot learning pipelines increasingly combine imitation learning pretraining with reinforcement learning fine-tuning, our results suggest that the optimal gain setting may need to change across stages—or that gain-aware curriculum design could improve end-to-end performance. Our framework also raises questions for adjacent areas. Modern humanoid robots increasingly use RL-trained whole-body tracking policies as low-level controllers, analogous to the PD controllers studied here. Yet how their compliance shapes high-level policy learning remains unexplored. Similarly, paradigms that learn manipulation skills from human videos [21, 22] or wearable devices [23] typically treat observed next timestep state as the action label, implicitly assuming perfect target tracking, which our results suggest may be suboptimal for behavior cloning. Whether these gain-dependent trends generalize to such cross-embodiment or whole-body control settings remains an open question, and we hope our findings offer a useful lens for investigating these directions.

References

[1] M. Takegaki and S. Arimoto, "A new feedback method for dynamic control of manipulators," 1981.

[2] R. Kelly, "Pd control with desired gravity compensation of robotic manipulators: a review," *The International Journal of Robotics Research*, vol. 16, no. 5, pp. 660–672, 1997.

[3] E. Aljalbout, F. Frank, M. Karl, and P. van der Smagt, "On the role of the action space in robot manipulation learning and sim-to-real transfer," *IEEE Robotics and Automation Letters*, vol. 9, no. 6, p. 5895–5902, Jun. 2024. [Online]. Available: http://dx.doi.org/10.1109/LRA.2024.3398428

[4] D. Kim, G. Berseth, M. Schwartz, and J. Park, "Torque-based deep reinforcement learning for task-and-robot agnostic learning on bipedal robots using sim-to-real transfer," *IEEE Robotics and Automation Letters*, vol. 8, no. 10, p. 6251–6258, Oct. 2023. [Online]. Available: http://dx.doi.org/10.1109/LRA.2023.3304561

[5] J. Eßer, G. B. Margolis, O. Urbann, S. Kerner, and P. Agrawal, "Action space design in reinforcement learning for robot motor skills," in *8th Annual Conference on Robot Learning*, 2024.

[6] Y. Wu, F. Zhao, T. Tao, and A. Ajoudani, "A framework for autonomous impedance regulation of robots based on imitation learning and optimal control," *IEEE Robotics and Automation Letters*, vol. 6, no. 1, pp. 127–134, 2021.

[7] K. Kronander and A. Billard, "Learning compliant manipulation through kinesthetic and tactile human-robot interaction," *IEEE Transactions on Haptics*, vol. 7, no. 3, pp. 367–380, 2014.

[8] G. B. Margolis, M. Wang, N. Fey, and P. Agrawal, "Soft-mimic: Learning compliant whole-body control from examples," *arXiv preprint arXiv:2510.17792*, 2025.

[9] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," 2017. [Online]. Available: https://arxiv.org/abs/1703.06907

[10] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2018, p. 3803–3810. [Online]. Available: http://dx.doi.org/10.1109/ICRA.2018.8460528

[11] OpenAI, I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, J. Schneider, N. Tezak, J. Tworek, P. Welinder, L. Weng, Q. Yuan, W. Zaremba, and L. Zhang, "Solving rubik's cube with a robot hand," 2019. [Online]. Available: https://arxiv.org/abs/1910.07113

[12] F. Muratore, F. Ramos, G. Turk, W. Yu, M. Gienger, and J. Peters, "Robot learning from randomized simulations: A review," 2022. [Online]. Available: https://arxiv.org/abs/2111.00956

[13] A. Khazatsky, K. Pertsch, S. Nair, A. Balakrishna, S. Dasari, S. Karamcheti, S. Nasiriany, M. K. Srirama, L. Y. Chen, K. Ellis *et al.*, "Droid: A large-scale in-the-wild robot manipulation dataset," *arXiv preprint arXiv:2403.12945*, 2024.

[14] Q. Vuong, S. Levine, H. R. Walke, K. Pertsch, A. Singh, R. Doshi, C. Xu, J. Luo, L. Tan, D. Shah *et al.*, "Open x-embodiment: Robotic learning datasets and rt-x models," in *Towards Generalist Robots: Learning Paradigms for Scalable Skill Acquisition@ CoRL2023*, 2023.

[15] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," *The International Journal of Robotics Research*, vol. 44, no. 10-11, pp. 1684–1704, 2025.

[16] Y. Park, G. B. Margolis, and P. Agrawal, "Automatic environment shaping is the next frontier in rl," *arXiv preprint arXiv:2407.16186*, 2024.

[17] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.

[18] A. Serrano-Muñoz, D. Chrysostomou, S. Bøgh, and N. Arana-Arexolaleiba, "skrl: Modular and flexible library for reinforcement learning," *Journal of Machine Learning Research*, vol. 24, no. 254, pp. 1–9, 2023. [Online]. Available: http://jmlr.org/papers/v24/23-0112.html

[19] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017. [Online]. Available: https://arxiv.org/abs/1707.06347

[20] NVIDIA, :, M. Mittal, P. Roth, J. Tigue, A. Richard, O. Zhang, P. Du, A. Serrano-Muñoz, X. Yao, R. Zurbrügg, N. Rudin, L. Wawrzyniak, M. Rakhsha, A. Denzler, E. Heiden, A. Borovicka, O. Ahmed, I. Akinola, A. Anwar, M. T. Carlson, J. Y. Feng, A. Garg, R. Gasoto, L. Gulich, Y. Guo, M. Gussert, A. Hansen, M. Kulkarni, C. Li, W. Liu, V. Makoviychuk, G. Malczyk, H. Mazhar, M. Moghani, A. Murali, M. Noseworthy, A. Poddubny, N. Ratliff, W. Rehberg, C. Schwarke, R. Singh, J. L. Smith, B. Tang, R. Thaker, M. Trepte, K. V. Wyk, F. Yu, A. Millane, V. Ramasamy, R. Steiner, S. Subramanian, C. Volk, C. Chen, N. Jawale, A. V. Kuruttukulam, M. A. Lin, A. Mandlekar, K. Patzwaldt, J. Welsh, H. Zhao, F. Anes, J.-F. Lafleche, N. Moënne-Loccoz, S. Park, R. Stepinski, D. V. Gelder, C. Amevor, J. Carius, J. Chang, A. H. Chen, P. de Heras Ciechomski, G. Daviet, M. Mohajerani, J. von Muralt, V. Reutskyy, M. Sauter, S. Schirm, E. L. Shi, P. Terdiman, K. Vilella, T. Widmer, G. Yeoman, T. Chen, S. Grizan, C. Li, L. Li, C. Smith, R. Wiltz, K. Alexis, Y. Chang, D. Chu, L. J. Fan, F. Farshidian, A. Handa, S. Huang, M. Hutter, Y. Narang, S. Pouya, S. Sheng, Y. Zhu, M. Macklin,

A. Moravanszky, P. Reist, Y. Guo, D. Hoeller, and G. State, "Isaac lab: A gpu-accelerated simulation framework for multi-modal robot learning," 2025. [Online]. Available: https://arxiv.org/abs/2511.04831

[21] R.-Z. Qiu, S. Yang, X. Cheng, C. Chawla, J. Li, T. He, G. Yan, D. J. Yoon, R. Hoque, L. Paulsen *et al.*, "Humanoid policy˜ human policy," *arXiv preprint arXiv:2503.13441*, 2025.

[22] K. Grauman, A. Westbury, E. Byrne, Z. Chavis, A. Furnari, R. Girdhar, J. Hamburger, H. Jiang, M. Liu, X. Liu *et al.*, "Ego4d: Around the world in 3,000 hours of egocentric video," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 18 995–19 012.

[23] C. Chi, Z. Xu, C. Pan, E. Cousineau, B. Burchfiel, S. Feng, R. Tedrake, and S. Song, "Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots," *arXiv preprint arXiv:2402.10329*, 2024.

[24] Y. Park and P. Agrawal, "Using apple vision pro to train and control robots," 2024. [Online]. Available: https://github.com/Improbable-AI/VisionProTeleop

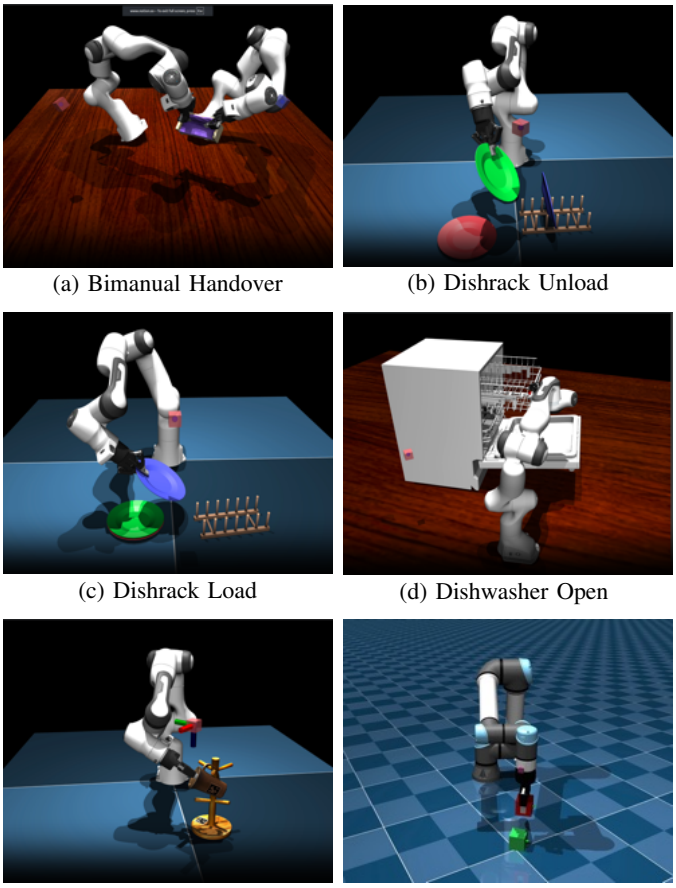
[25] M. Nomura and M. Shibata, "cmaes : A simple yet practical python library for cma-es," 2024. [Online]. Available: https://arxiv.org/abs/2402.01373

[26] I. A. Lab, "aiofranka: Asyncio-based franka robot control," 2025. [Online]. Available: https://github.com/Improbable-AI/aiofranka

Appendix A
Experiments

A. Behavior Cloning

1) Task Descriptions: The six tasks we study are: Bimanual Handover, Dishrack Unload, Dishrack Load, Dishwasher Open, Mug Hang, and Block Stack (Figure 11). For all tasks besides Block Stack, we collect 100 teleoperated demonstrations with the Apple Vision Pro [24] for each task. For Block Stack, we use motion-planned trajectories. These demonstrations are collected at 500Hz recording raw torques generated from operational space controller, and are retargeted to joint-level position targets for each gain setting at 50Hz for gain-dependent policy learning.

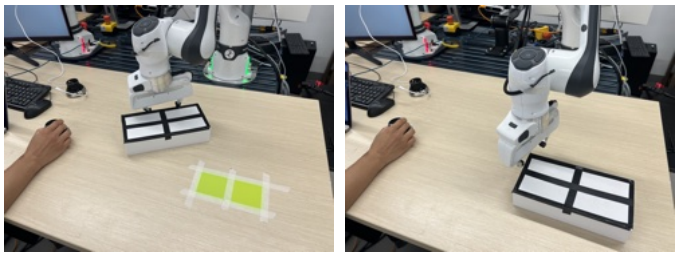


(a) Bimanual Handover (b) Dishrack Unload (c) Dishrack Load (d) Dishwasher Open (e) Mug Hang (f) Block Stack

Fig. 11: Six tasks used for behavior cloning.

2) Nominal Training Configuration: As a nominal configuration, we use VAE as a generative model with MLP network with observation size 10 and action chunk size 10, with privileged simulation states as inputs, using absolute joint as action space.

3) Ablation Training Configurations: We present ablation results across dataset size (Figure 18), policy architectures (Figure 19), action chunk size (Figure 20), action representation (Figure 21), and control frequency (Figure 22). Across all ablations, we observe a similar preference for compliant and overdamped gain regimes.



(a) Task In-Progress (b) Task Complete

Fig. 12: **Non-prehensile box manipulation task for the user study.** A single trial of the task involves teleoperating the robot from a reset pose to make contact with the box, then pushing the box towards the goal. The task is complete when the green square is completely occluded by the box (b).

4) Scaling Law: Beyond absolute performance, the choice of controller gains also affects how efficiently policies improve with additional data. As shown in Fig. 23, compliant and overdamped gains exhibit steeper scaling with dataset size, implying that data collection efforts yield greater returns in this regime. For practitioners with limited demonstration budgets, this makes gain selection a critical lever for maximizing policy performance.

B. User Study

1) Task Description: We design a non-prehensile box manipulation task for the user study (Figure 7). This task is contact-rich and requires precision, but is still largely achievable even with unintuitive controller gain configurations. For each trial, users teleoperate a Franka Research 3 Robot with a SpaceMouse in order to push the box from an initial pose to the goal (Figure 12b). The box is always initialized to the left and off-axis relative to the goal (Figure 12a), but the precise pose is random. The goal pose is fixed in every trial.

2) Experimental Design and Results: The user study contains 1,297 total trials collected by 12 users. We record both successful and failed trials. Trials can fail if (a) the robot faults by hitting a position, velocity, or torque limit, or (b) the user fails to complete the task by pushing the box off the table or out of the workspace of the robot. Each user completes trials for 1 hour, where the robot gain setting is randomly sampled for each trial. We randomize the order in which gain settings are presented to participants because users, particularly novice users, tend to improve at the task across all gain settings over the course of the hour session. For each trial, we record whether it was successful, the time to completion, and the user's subjective rating of the gain setting. The subjective rating is on a scale from 1-5, where 1 means the gain setting is a completely unintuitive interface, and 5 means it is a completely intuitive interface. Users complete the survey in Figure 13 after each trial to record their subjective rating. The results are presented in Figure 14.

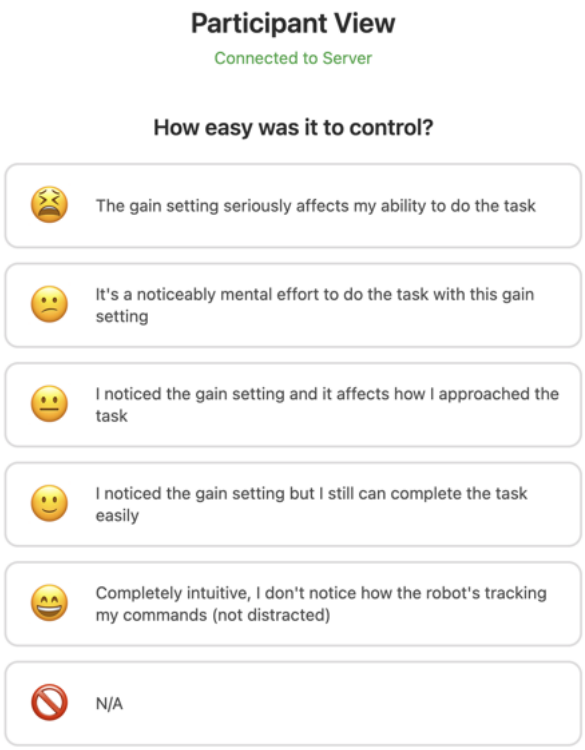


Fig. 13: **User study survey.** After each trial, users complete the survey to rate their subjective experience teleoperating with a given gain setting.

C. Online Reinforcement Learning

1) Task Descriptions: The five tasks we study are: FR3 Joint-Reach, FR3 EE-Reach, FR3 Lift Cube, FR3 Open Drawer, and Unitree G1 Track Velocity (Figure 15). Each task is derived from the IsaacLab [20] template environments.

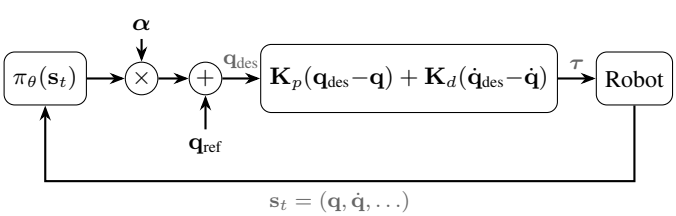


Fig. 16: **Action representation.** The policy output is scaled by a per-joint-group vector $\boldsymbol{\alpha}$ and added to a reference position $\mathbf{q}_{\text{ref}}$ to produce the position target $\mathbf{q}_{\text{des}}$ sent to the PD controller.

2) Action Representations: For all tasks, the position target sent to the low-level PD controller at each timestep is:

$$\mathbf{q}_{\text{des}}(t) = \boldsymbol{\alpha} \odot \pi_\theta(\mathbf{s}_t) + \mathbf{q}_{\text{ref}}(t) \tag{6}$$
$$\boldsymbol{\alpha} = [\underbrace{\alpha_1, \ldots, \alpha_1}_{\mathcal{G}_1}, \underbrace{\alpha_2, \ldots, \alpha_2}_{\mathcal{G}_2}]$$

where $\mathbf{q}_{\text{ref}}(t)$ is an offset equal to either the current joint position $\mathbf{q}(t)$ or the default joint position $\mathbf{q}_0$, depending on the task. Joints are partitioned into two groups, $\mathcal{G}_1$ and $\mathcal{G}_2$, with shared scale factors α_1 and α_2 respectively (Table II). Both scale factors are tuned via computational hyperparameter optimization [17] to adapt the action space to each gain setting.

TABLE II: Action representation across RL tasks.

Task	$\mathbf{q}_{\text{ref}}(t)$	$\mathcal{G}_1$	$\mathcal{G}_2$	Gripper
FR3 Joint-Reach	$\mathbf{q}(t)$	$q_{0\text{–}3}$ (elbow)	$q_{4\text{–}6}$ (wrist)	–
FR3 EE-Reach	$\mathbf{q}(t)$	$q_{0\text{–}3}$ (elbow)	$q_{4\text{–}6}$ (wrist)	–
FR3 Lift Cube	$\mathbf{q}(t)$	$q_{0\text{–}3}$ (elbow)	$q_{4\text{–}6}$ (wrist)	binary
FR3 Open Drawer	$\mathbf{q}(t)$	$q_{0\text{–}3}$ (elbow)	$q_{4\text{–}6}$ (wrist)	binary
G1 Locomotion	$\mathbf{q}_0$	$q_{0\text{–}13}$ (lower body)	$q_{13\text{–}36}$ (upper body)	–

For tasks with a gripper (Table II), the policy outputs an additional continuous value that is thresholded at zero, commanding the fingers to either fully opened (0.04 m) or fully closed (0.0 m). The gripper joint gains are held fixed across all experiments.

3) Success Criteria: For each policy trained during hyperparameter optimization, we record the success rate across 100 simulated trials according to the success metrics in Table III. We evaluate the best (highest reward) checkpoint for each policy.

TABLE III: Success criteria for each RL task.

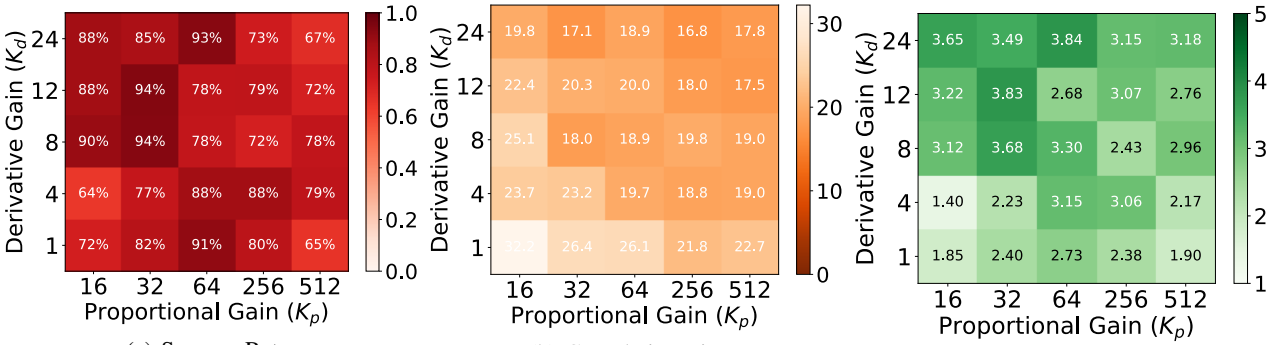
Task	Criterion	Threshold
FR3 Joint-Reach	$\lVert\mathbf{q} - \mathbf{q}_{\text{goal}}\rVert < \epsilon$	$\epsilon = 0.1$ rad
FR3 EE-Reach	$\lVert\mathbf{p} - \mathbf{p}_{\text{goal}}\rVert < \epsilon_p$, $\lVert\Delta\theta\rVert < \epsilon_r$	$\epsilon_p = 0.02$ m, $\epsilon_r = 0.1$ rad
FR3 Lift Cube	$\lVert\mathbf{p}_{\text{obj}} - \mathbf{p}_{\text{goal}}\rVert < \epsilon$	$\epsilon = 0.05$m
FR3 Open Drawer	$d_{\text{drawer}} > d_{\text{min}}$	$d_{\text{min}} = 0.2$
G1 Locomotion	$\lVert\dot{\mathbf{q}}\rVert / \lVert\dot{\mathbf{q}}_{\text{goal}}\rVert > \rho$	$\rho = 0.4$

In order to adapt the environment to new gain settings, we leverage computational hyperparameter optimization with Optuna to tune the action scales and, for the FR3 EE-Reach task, the reward term weights. We use the TPE optimizer with default hyperparameters, where the objective function is the task success rate.

4) PPO Hyperparameters: We use largely the same PPO hyperparameters as the IsaacLab [20] template environments. Hyperparameters, including any changes we made, are reproduced here (Table IV and Table V).

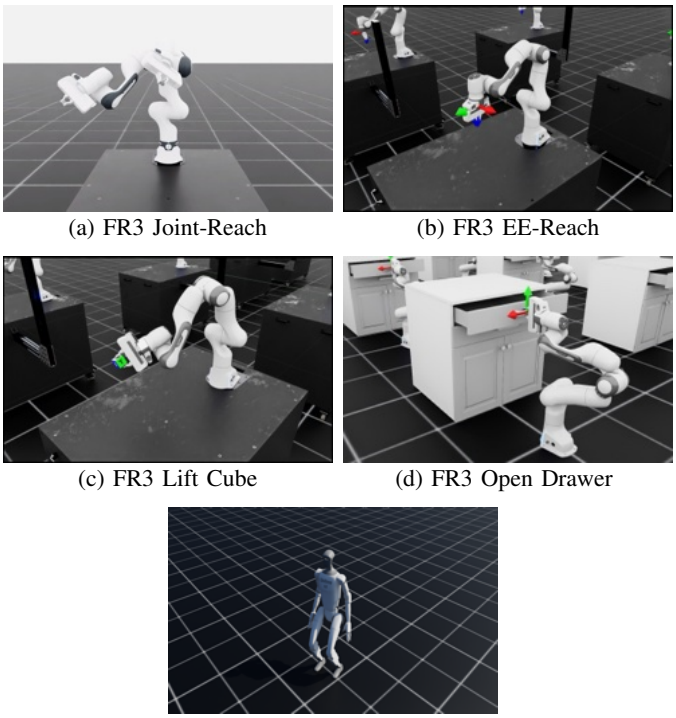
TABLE IV: PPO hyperparameters shared across all tasks.

Hyperparameter	Value
Algorithm	PPO (SKRL)
Discount factor γ	0.99
GAE λ	0.95
Learning epochs	5
Clip range (ratio)	0.2
Clip range (value)	0.2
Grad norm clip	1.0
LR scheduler	KL-Adaptive
Activation	ELU
Seed	42



(a) Success Rate (b) Completion Time (c) User Rating

Fig. 14: **Teleoperation performance under different gain regimes.** Compliant and overdamped controllers (grid top-left) achieve similar or better success rates, user ratings, and completion times to stiffer settings.



(a) FR3 Joint-Reach (b) FR3 EE-Reach (c) FR3 Lift Cube (d) FR3 Open Drawer (e) Unitree G1 Track Velocity

Fig. 15: **Five tasks for online RL solution existence proof.** For each task, we trained a successful policy for 8+ gain configurations spanning the range of stiff / compliant, overdamped / underdamped.

TABLE V: PPO hyperparameters that vary across tasks.

Hyperparameter	Reach	Lift	Drawer	G1
Network layers	[64, 64]	[256, 128, 64]	[256, 128, 64]	[256, 128, 128]
Learning rate	1e-3	1e-3	5e-4	1e-3
KL threshold	0.01	0.01	0.008	0.01
Rollouts	24	24	96	24
Mini-batches	4	4	96	4
Entropy coeff.	0.01	0.01	0.001	0.008
Value loss coeff.	1.0	1.0	2.0	1.0
Min log std	-3.0	-3.0	-20.0	-20.0
State preprocessor	RSS	RSS	–	–
Value preprocessor	RSS	RSS	–	–
Timesteps	24k	48k	38.4k	12k

RSS = RunningStandardScaler.

D. Sim2Real

1) System Identification Data Collection: For each gain configuration $(\mathbf{K}_p, \mathbf{K}_d)$, the real robot executes a sinusoidal reference trajectory $\mathbf{q}_{\text{des}}(t) = \mathbf{q}_0 + 0.1\sin(\pi t/50)$ applied uniformly across all joints for 4 seconds. During execution, we log joint positions $\mathbf{q}$, joint velocities $\dot{\mathbf{q}}$, and desired positions $\mathbf{q}_{\text{des}}$ at 50 Hz. The low-level torque controller on the real robot runs at 1 kHz.

To match the real-world setup, the IsaacLab simulation environment updates position commands at 50 Hz with a physics simulation rate of 100 Hz. We use 100 Hz rather than 1 kHz physics to keep RL training times tractable. We note that this fidelity gap between the real robot's 1 kHz control loop and the simulator's 100 Hz physics rate contributes to the sim-to-real discrepancy that system identification aims to minimize.

2) System Identification Procedure: For each gain configuration, we use CMA-ES [25] to optimize simulation parameters per-actuator ψ (Table VI) to minimize the discrepancy between real and simulated response trajectories.

TABLE VI: System identification parameter bounds. Parameters are optimized per-actuator.

Parameter	Lower	Upper
Stiffness K_p	1	1024
Damping K_d	1	1024
Armature	0	0.5
Static friction	0.01	1.0
Dynamic friction ratio	0	1.0
Viscous friction	0	1.0

The objective function is the sum of spectral MSE losses for joint positions and velocities:

$$\mathcal{L}(\psi) = \mathcal{L}_{\text{spec}}(\mathbf{q}^{\text{real}}, \mathbf{q}^{\text{sim}}(\psi)) + \mathcal{L}_{\text{spec}}(\dot{\mathbf{q}}^{\text{real}}, \dot{\mathbf{q}}^{\text{sim}}(\psi)) \quad (7)$$

where $\mathcal{L}_{\text{spec}}$ computes the mean squared error between the discrete Fourier transforms of the simulated and real trajectories. Matching in the frequency domain encourages the optimizer to capture oscillatory behavior and damping characteristics.

CMA-ES runs for 200 iterations with an initial step size of $\sigma = 3.0$, independently for all 49 gain configurations. We

visualize the system identification result against the real-world trajectory for four gain settings in Figure 17.

3) Training Deployable Policies: We train deployable FR3 Joint-Reach and FR3 EE-Reach policies. To discover policies that respect the real robot's limits, we modify the outer-loop Optuna objective to a two-stage formulation that always prefers constraint-satisfying configurations over violating ones:

$$\mathcal{J} = \begin{cases} 1 + r_{\text{success}} & \text{if all } v_c \leq \bar{v}_c \\ r_{\text{success}} \prod_{c \in \mathcal{C}} \phi_c & \text{otherwise} \end{cases} \tag{8}$$

where v_c and $\bar{v}_c$ are the violation rate and allowed threshold for each constraint $c \in \mathcal{C} = \{\text{position, velocity, torque, torque rate}\}$, and the penalty terms are:

$$\phi_c = \begin{cases} 1 & \text{if } v_c \leq \bar{v}_c \\ \max\big(0,\ 1 - (v_c - \bar{v}_c)\big) & \text{otherwise} \end{cases} \tag{9}$$

Since $\mathcal{J} \in [1, 2]$ when all constraints are met and $\mathcal{J} \in [0, 1)$ otherwise, feasible configurations are always ranked above infeasible ones. Within each regime, higher success rate is favored.

We set $\bar{v}_c = 0$ for position, velocity, and torque constraints, requiring zero violations. For torque rate, we allow $\bar{v}_c = 0.2$, since the real robot enforces torque rate limiting at 1 kHz as a hardware safety layer; as long as the learned policy does not rely on frequent high-rate torque switching, occasional violations in simulation are acceptable.

E. Real-World Deployment

We deploy learned policies on a Franka FR3 robot using the `aiofranka`[26] library, which provides an asynchronous interface for real-time torque control. The deployment system consists of two nested control loops:

- **Inner loop (1 kHz):** A joint-space impedance controller computes torques as:

$$\boldsymbol{\tau} = \mathbf{K}_p(\mathbf{q}_{des} - \mathbf{q}) - \mathbf{K}_d\dot{\mathbf{q}} \tag{10}$$

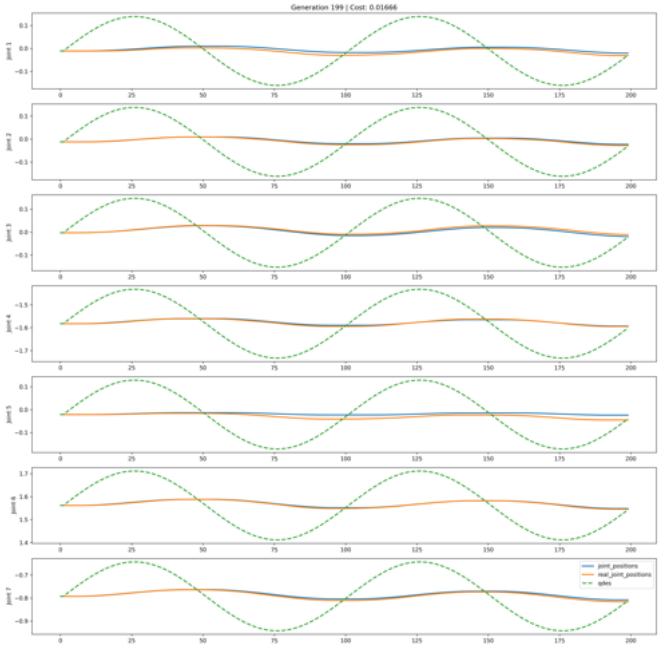
  where $\mathbf{q}_{des} \in \mathbb{R}^7$ is the commanded joint position, $\mathbf{q}, \dot{\mathbf{q}} \in \mathbb{R}^7$ are the current joint positions and velocities, and $\mathbf{K}_p, \mathbf{K}_d \in \mathbb{R}^{7\times 7}$ are diagonal stiffness and damping gain matrices. Before commanding the robot, torques are clamped to the torque limits, and torque rates are limited to $|\dot{\tau}_i| \leq 990$ Nm/s.
- **Outer loop (50 Hz):** The learned policy outputs actions at 50 Hz, which are converted to position setpoints $\mathbf{q}_{des}$ for the inner impedance controller.

F. Sim-to-Real Analysis

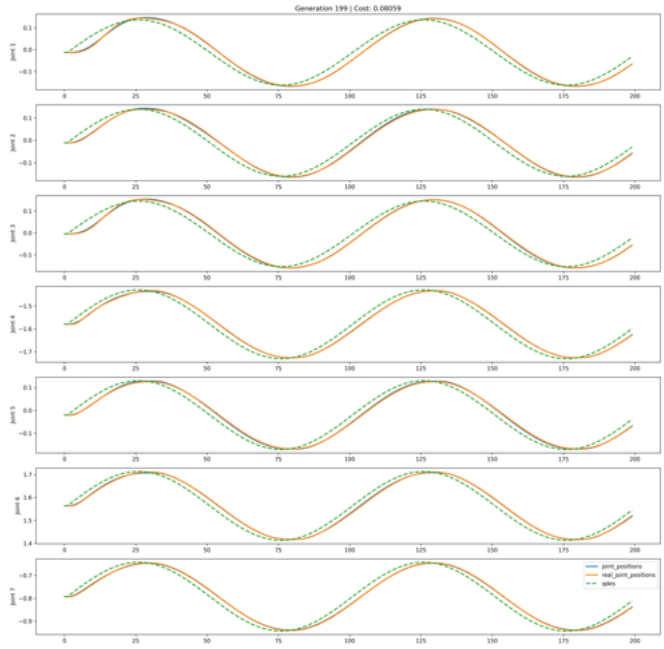
We compute the NN error as:

$$\mathcal{E}_{\text{NN}} = \text{RMS}\left(\pi_\theta(\mathbf{s}_t^{\text{real}}) - \pi_\theta(\mathbf{s}_t^{\text{sim}})\right) \tag{11}$$

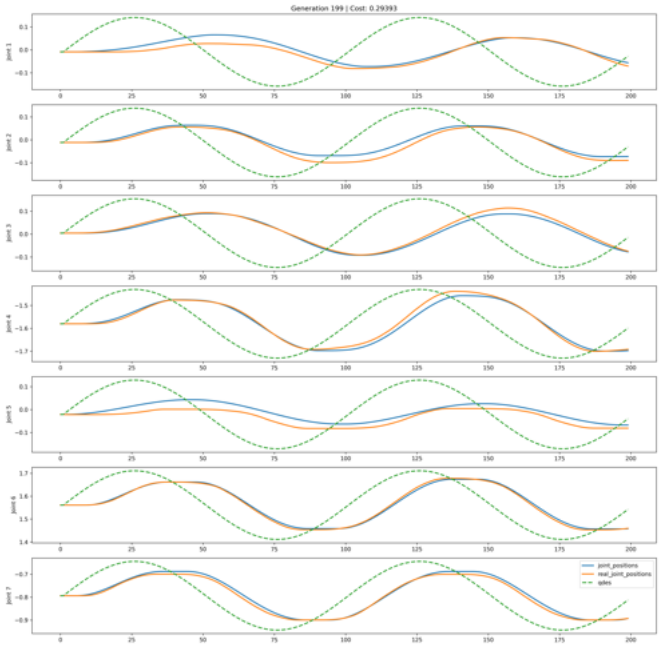
This measures the trajectory-wise difference between the policy's outputs in simulation and on the real robot, when the initial and goal configurations are matched. The NN error for each of our reaching tasks (Figure 25) is well correlated with the trajectory error (Figure 24), suggesting that the sim-to-real gap is primarily caused by the policy receiving out-of-distribution states on the real robot, rather than instability in the low-level controller.



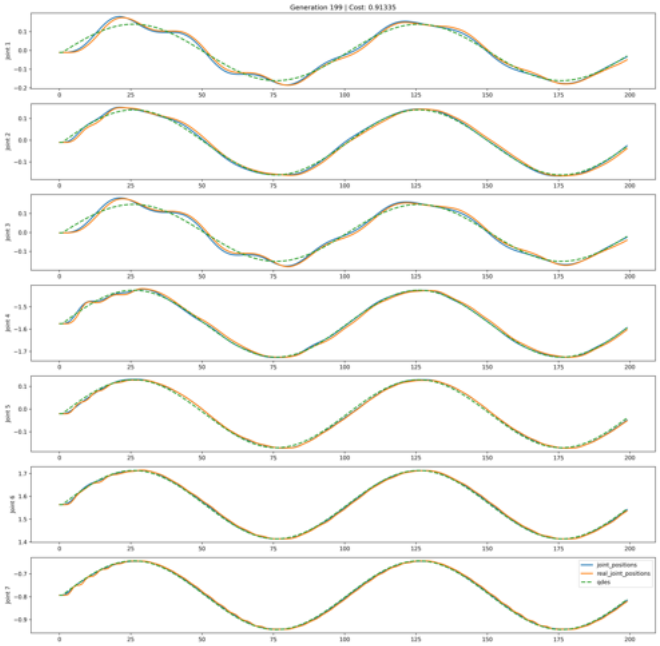
(a) Compliant, Overdamped ($K_p = 16,\ K_d = 24$)



(b) Stiff, Overdamped ($K_p = 512,\ K_d = 24$)

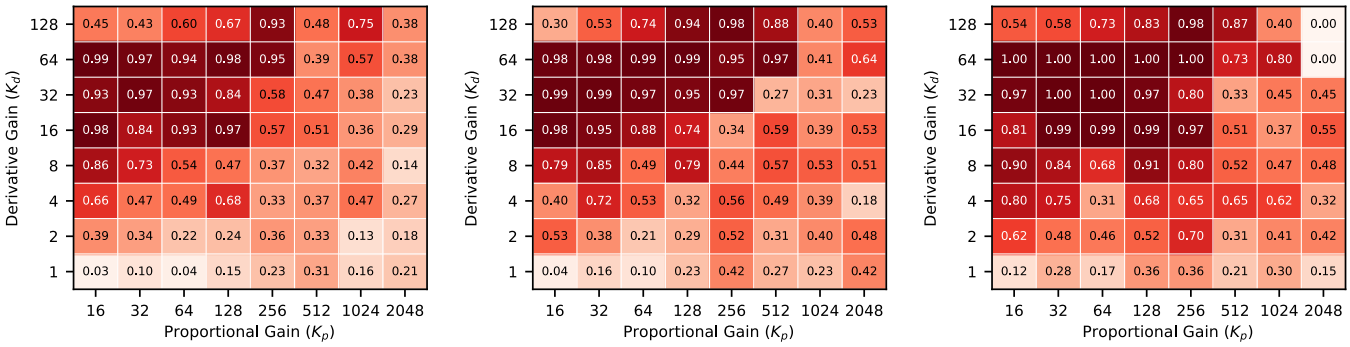


(c) Compliant, Underdamped ($K_p = 16,\ K_d = 2$)



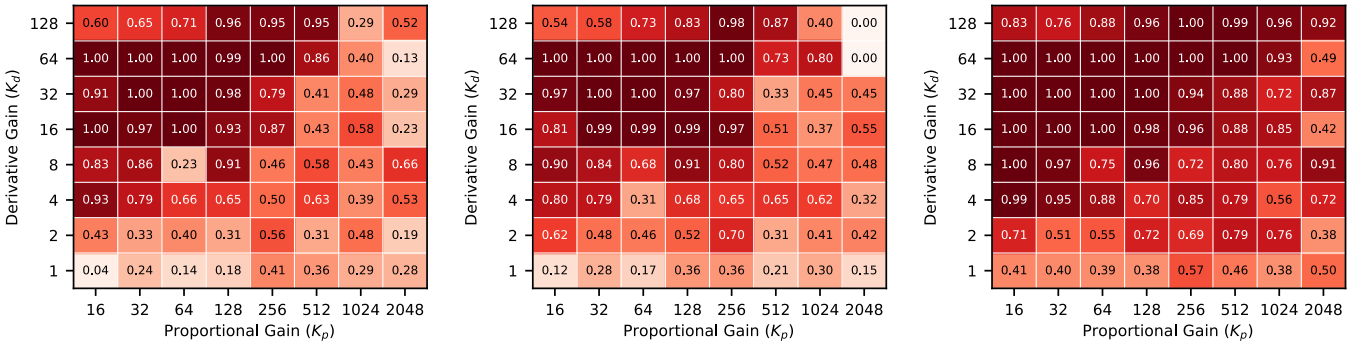
(d) Stiff, Underdamped ($K_p = 512,\ K_d = 2$)

Fig. 17: **System identification result for sample gain settings in each gain regime.** We show commanded positions (green), real-world achieved positions (orange), and simulation positions (blue) achieved with the optimal actuator parameters.



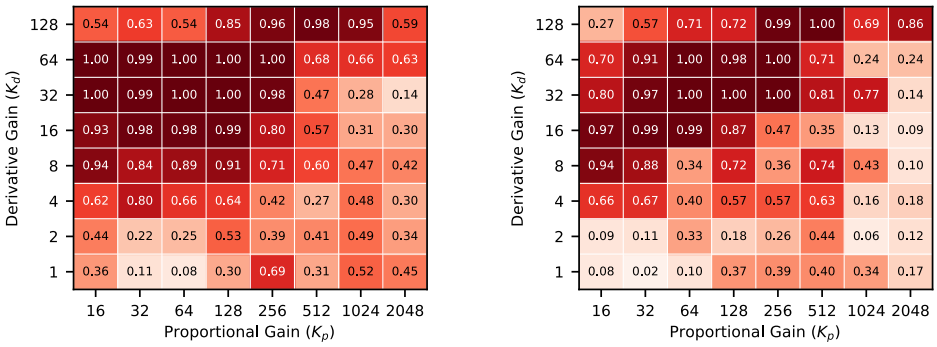
(a) 50 Trajectories (b) 100 Trajectories (c) 900 Trajectories

Fig. 18: **Behavior cloning performance across dataset size.** Success rate per gain setting for the Block Stack task. The preference for compliant and overdamped gain settings is maintained across dataset sizes (a-c).



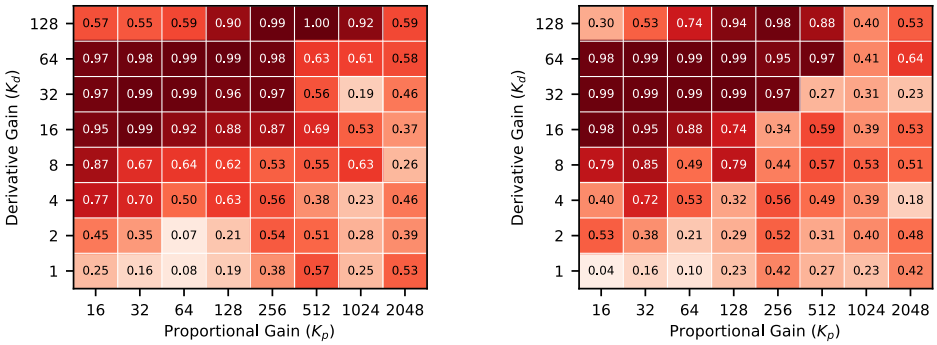
(a) Regression (b) VAE (c) Diffusion Policy

Fig. 19: **Behavior cloning performance across policy architectures.** Success rate per gain setting for the Block Stack task. The preference for compliant and overdamped gain settings is maintained across policy architectures (a-c).



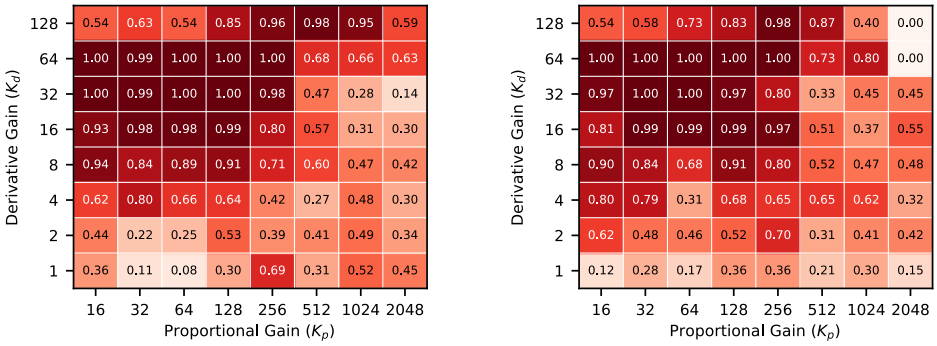
(a) No Action Chunking (b) Action Chunk Size 10

Fig. 20: **Behavior cloning performance across action chunk size.** Success rate per gain setting for the Block Stack task. The preference for compliant and overdamped gain settings is observed when predicting both single actions (a) and action chunks (b).



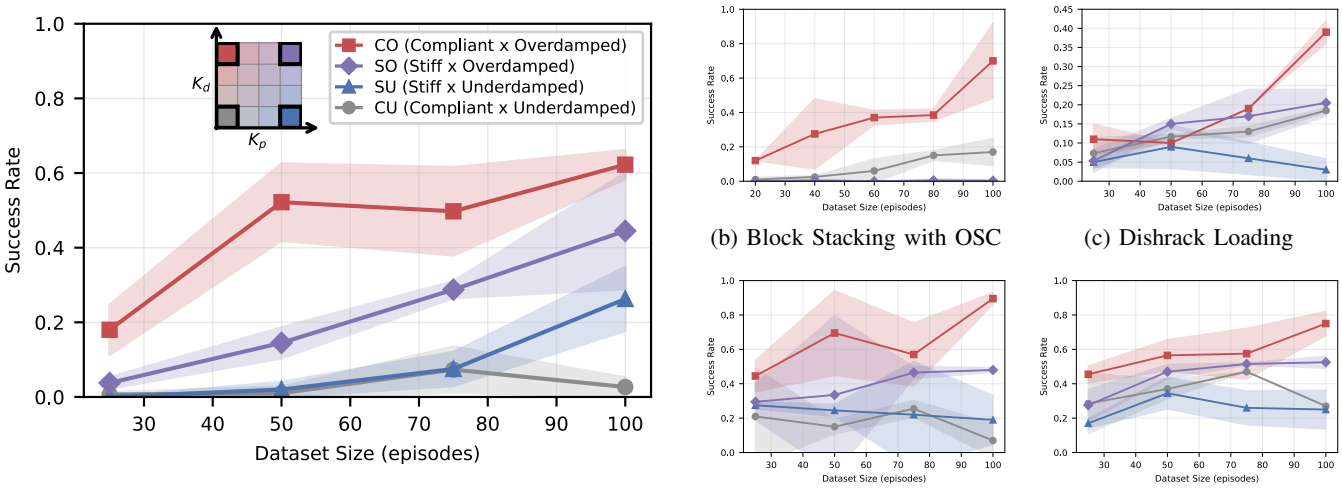
(a) Absolute Joint Action (b) Delta Joint Action

Fig. 21: **Behavior cloning performance across action representations.** Success rate per gain setting for the Block Stack task. The preference for compliant and overdamped gain settings is observed when predicting both absolute (a) and relative (b) joint position actions.



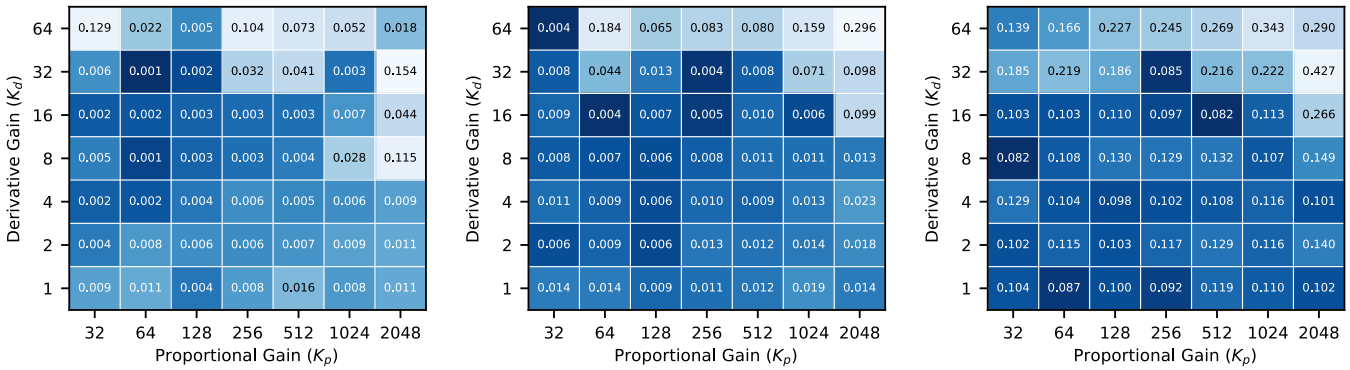
(a) 10Hz Control Frequency (b) 50Hz Control Frequency

Fig. 22: **Behavior cloning performance across control frequencies.** Success rate per gain setting for the Block Stack task. The preference for compliant and overdamped gain settings is observed when predicting actions at 10Hz (a) and 50Hz (b).



(a) Block Stacking with UR (b) Block Stacking with OSC (c) Dishrack Loading (d) Dishwasher Opening (e) Mug Hanging

Fig. 23: **Offline imitation learning scales more favorably under compliant and overdamped gains.** Success rate as a function of dataset size across tasks and robot embodiments. Policies trained with low stiffness and high damping achieve higher success with fewer demonstrations, while stiff or weakly damped controllers exhibit poorer data scaling.

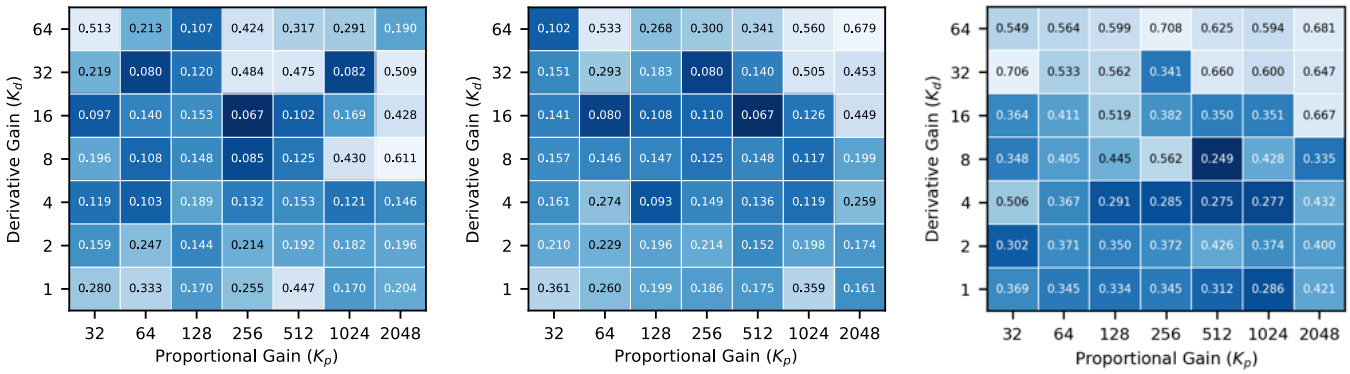


(a) Joint-Reach Sim2Real trajectory error without Domain Randomization

(b) Joint-Reach Sim2Real trajectory error with Domain Randomization

(c) EE-Reach Sim2Real trajectory error without Domain Randomization

Fig. 24: **Stiff and overdamped gain settings reduce sim2real transferability.** The Sim2Real trajectory error (Eq.5) is consistently larger (light blue) in the stiff and overdamped regime (a-c).



(a) Joint-Reach Sim2Real NN error without Domain Randomization

(b) Joint-Reach Sim2Real NN error with Domain Randomization

(c) EE-Reach Sim2Real NN error without Domain Randomization

Fig. 25: **Stiff and overdamped gains increase Sim2Real NN error.** The Sim2Real NN error (Eq.11) is consistently larger (light blue) in the stiff and overdamped regime (a-c).